



#15

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Applicants:	Tinku Acharya et al.	§	Art Unit:	2613
		§		
Serial No.:	09/722,988	§		
		§	Examiner:	Y. Young Lee
Filed:	November 27, 2000	§		
		§		
For:	Wavelet Coding of Video	§	Atty. Docket No.:	ITL.0514US (P9822)
		§		
Customer No.:	21906	§	Confirmation No.:	5871

Mail Stop Petition
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Applicant respectfully appeals from the final rejection mailed November 7, 2003, finally rejecting claims 1-30.

I. REAL PARTY IN INTEREST

The real parties in interest are the assignees Intel Corporation and the Indian Institute of Technology, the assignees by virtue of the assignments recorded at Reel/Frame 011320/0967 and 011358/0951 respectively.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF THE CLAIMS

Claims 1-30 are all original claims. Claims 1-30 have been finally rejected and are the subject of this appeal.

05/18/2004 SLUANG1 00000076 09722988

02 FC:1402

330.00 OP

Date of Deposit:	5.13.2004
I hereby certify under 37 CFR §1.8(a) that this correspondence is being deposited with the United States Postal Service as First Class Mail with sufficient postage on the date indicated above and is addressed to: Mail Stop Petition, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.	
Rebecca R. Ginn	

IV. STATUS OF AMENDMENTS

There are no unentered amendments.

V. SUMMARY OF THE INVENTION

Referring to Figure 2, an embodiment 119 of a compression program, in accordance with one embodiment of the invention, may enable a processor 112 to encode wavelet coefficients in a bit-wise fashion in a technique which may be called modified embedded zerotree (MEZT) coding. In this manner, instead of classifying the wavelet coefficients (as zerotree roots or isolated zeros, as examples), the processor 112 may produce codes to classify the bits of the wavelet coefficients. For example, in some embodiments, the processor 112 may classify a particular bit as being either a zerotree root, an isolated zero, a positive node or a negative node. Unlike conventional zerotree coding schemes, thresholds are not computed to identify insignificant values, as the “0” bit is treated as being insignificant and the “-1” and “1” bits are treated as being significant. Specification, p. 4, lines 21-31.

In this manner, the processor 112 may generate one of the following codes to classify a particular bit: a “P” code to indicate a positive node if the bit indicates a “1”; an “N” code to indicate a negative node if the bit indicates a “-1”; an “R” code to indicate that a “0” bit is a zerotree root; and an “IZ” code to indicate that a “0” bit is an isolated zero. In some embodiments, a particular bit is classified as a negative node only if the bit is the most significant nonzero bit and the bit indicates a “-1.” For example, for a coefficient of “-3” that is represented by the three bits “-011,” the processor 112 generates an N code to represent the middle bit. However, for this example, the processor 112 generates a P code to represent the least significant bit. Specification, p. 5, lines 1-9.

For purposes of providing the wavelet coefficients, the processor 112 may, via wavelet transformations, decompose coefficients that represent pixel intensities of an original image. These wavelet coefficients, in turn, form subbands that are located in multiple decomposition levels. To classify the bits, the processor 112, in some embodiments, may execute the program 119 to process the bits based on their associated bit position, or order. In this manner, the bits of each bit order form a hierarchical tree that the processor 112 may traverse to classify each of the bits of the tree as being either a zerotree root, an isolated zero, a negative node or a positive node. Thus, as an example, the most significant bits of the wavelet coefficients (this bit may also be zero) are associated with one hierarchical tree (and one bit order), and the next most significant bits are associated with another hierarchical tree (and another bit order). Specification, p. 5, lines 10-20.

For example, if the absolute maximum wavelet coefficient is represented by three bits (as an example), then all of the wavelet coefficients may be represented by three bits. Therefore, for this example, three hierarchical trees are formed. In this manner, the processor 112 produces a code for each bit based on its indicated value (i.e., “-1,” “0,” or “1”) and possibly (if the bit indicates a “0”) its position in the associated hierarchical tree. Specification, p. 5, lines 21-25.

In some embodiments, the processor 112 indicates the P, N, IZ and R codes via a bit stream that progressively indicates a more refined (i.e., a higher resolution) version of the original image over time. For example, the processor 112 may use the bits “00” to indicate the “P” code, the bits “01” to indicate the “N” code, the bits “10” to indicate the “R” code and the bits “11” to indicate the IZ code. Other coding schemes are possible. The progressive nature of the bit stream is attributable to the order in which the processor 112 processes the bit orders. For example, in some embodiments, the processor 112 may process the bit orders in a most

significant first fashion. Therefore, the processor 112 may initially produce code for all the bits that have the highest bit order, then produce code for all of the bits that have the next highest bit order, etc. As a result of this progressing coding, the resultant bit stream may initially indicate a coarser version of the original image. However, more refinements to the image are indicated by the bit stream over time, as the processor 112 produces the codes for the bits having the lower bit orders. Thus, in some embodiments, the resolution of the image that is indicated by the bit stream improves over time, a feature that may be desirable for bandwidth-limited systems. As a result, a decrease in resolution of the reconstructed image may be traded for a decrease in communication bandwidth. Specification, p. 5, line 26 through p. 6, line 11.

Referring to Figure 3, in some embodiments, the processor 112 process the bits of each order in a predefined sequence. For example, for a particular bit order, the processor 112 may begin with the highest decomposition level and produce codes for the bits of the highest decomposition level before proceeding to produce codes for the bits of the next highest decomposition level. The processor 112 produces code(s) for the bit(s) of the LL subband and, then for each decomposition level, produces code(s) for the bit(s) of the LH subband, subsequently, produces code(s) for the bit(s) of the HL subband and lastly, produces code(s) the bit(s) of the HH subband. Specification, p. 6, lines 12-19.

As an example, the wavelet coefficients produced by a two level decomposition may be arranged in a matrix 40 that is depicted in Figure 4. In this manner, the matrix 40 may be viewed as being subdivided into four quadrants 30a, 30b, 30c and 30d. The upper right 30b, lower left 30c and lower right 30d quadrants includes the coefficients for the LH, HL and HH subband images, respectively, of the first decomposition level. The coefficients for the LL, LH, HL and HH subband images of the second decomposition level are located in the upper left 32a, upper

right 32b, lower left 32c and lower right 32d quadrants of the upper left quadrant 30a. The coefficients produced by further decomposition may be arranged in a similar manner. For example, for a third level of decomposition, the upper left quadrant 32a includes the wavelet coefficients of the LL, LH, HL and HH subbands of the third decomposition level.

Specification, p. 6, lines 20-30.

If the coefficient matrix that indicates the pixel intensities for the original image is a 4X4 matrix, then the matrix 40 may be of the form that is depicted in Figure 5. In this manner, the LL, LH, HL and HH subband images of the second decomposition level each have one coefficient, represented by “A” (for the LL subband image), “B” (for the LH subband image), “C” (for the HL subband image) and “D” (for the HH subband image), respectively. As depicted in Figure 5, for the first decomposition level, the coefficients for the LH, HL and HH subband images are represented by the following respective matrices:

$$\begin{bmatrix} E_1 & E_2 \\ E_3 & E_4 \end{bmatrix}, \begin{bmatrix} F_1 & F_2 \\ F_3 & F_4 \end{bmatrix}, \begin{bmatrix} G_1 & G_2 \\ G_3 & G_4 \end{bmatrix}$$

It is noted that each coefficient of the second decomposition level (except A), is associated with at least four coefficients of the first decomposition level, i.e., each coefficient of the first decomposition level has at least four descendant coefficients in the second decomposition level. Therefore, each bit in the first decomposition level has at least four descendent coefficients in the second decomposition level. Specification, p. 7, lines 1-14.

For each bit order, the processor 112 may process the bits in the scanning sequence described above. If a particular bit indicates a “1” or a “-1,” then the processor 112 generates the P or N code and proceeds to process the next bit in the scanning sequence. However, if a particular bit indicates a “0,” then the processor 112 may trace the bit through its descendants to

determine if the bit is an isolated zero or a zerotree root. The coefficients in the LL subband are simply entropy encoded. Specification, p. 7, lines 15-20.

Therefore, the processor 112 begins the encoding by generating codes for the third order bits (i.e., the most significant bits, which may be zero also) of the coefficients. More particularly, to generate the codes for the third order bits, the processor 112 follows the path 28 (see Figure 5) and produces the appropriate code for the third bit of each coefficient along the path 28. If a particular bit indicates a "0," then the processor 112 evaluates the descendents of the bit to find isolated zeros and zero roots. The coding of the third order bits by the processor 112 produces the following codes (listed in the order of production): P,R,R,R. Subsequently, the processor 112 produces the codes for the second order bits (listed in order of production): IZ,IZ,N,R,IZ,P,IZ,IZ,IZ,P,IZ,IZ. Lastly, the processor 112 produces the codes for the first order bits (listed in order of production): IZ,P,IZ,R,P,IZ,IZ,P,IZ,P,IZ,P. As described above, the processor 112 may indicate the codes via a two bit coding scheme and transmit the codes as produced via a bit stream. Specification, p. 8, lines 5-17.

As an example, another processor 200 (see Figure 2) may use the bit stream to reconstruct the coefficient matrix that indicates the pixel intensities of the original image in the following manner. Before the decoding begins, the processor 200 first receives an indication from the processor 112 that three levels of coding (i.e., one level for each bit order) have been used. After obtaining this information, the processor 200 may reconstruct the original coefficient matrix using the codes in the order that the codes are produced. Specification, p. 8, lines 18-24.

Referring to Figure 7, to summarize, the compression program 119, when executed by the processor 112 may cause the processor 112 to perform the following procedure to produce the above-described coding. First, the processor 112 may express (block 72) a matrix of

decomposed coefficients in a signed binary representation. Next, the processor 112 may determine (block 74) the number of digits that are needed to represent the absolute value of the maximum wavelet coefficient. This processor 112 uses a variable (called n) that indicates the current bit order being processed by the processor 112. In this manner, the processor 112 uses a software loop to process the bits, one bit order at a time. To accomplish this, the processor 112 produces codes (block 76) for the bits of the current bit order the using the techniques described above. Subsequently, the processor 112 determines (diamond 78) whether the rate of transmitted bits may exceed a predetermined bit rate. If so, the processor 112 terminates the coding for the current image to comply with the predetermined bit rate. Otherwise, the processor 112 determines (diamond 80) if all bit orders have been processed, i.e., the processor 112 determines if n equals "1." If not, the processor 112 decrements (block 75) the order that is indicated by the n variable by one and proceeds to block 76 to traverse the loop another time to produce codes for the bits of another bit order. Otherwise, the coding is complete. Specification, p. 9, line 12 through p. 10, line 5.

Referring back to Figure 2, in some embodiments, the processor 112 may be part of a computer system 100. The computer system 100 may include a bridge, or memory hub 116, and the processor 112 and the memory hub 116 may be coupled to a host bus 114. The memory hub 116 may provide interfaces to couple the host bus 114, a memory bus 129 and an Accelerated Graphics Port (AGP) bus 111 together. The AGP is described in detail in the Accelerated Graphics Port Interface Specification, Revision 1.0, published on July 31, 1996, by Intel Corporation of Santa Clara, California. A system memory 118 may be coupled to the memory bus 129 and store the compression program 119. As described above, the compression program 119, when executed by the processor 112, may cause the processor 112 to provide wavelet

coefficients that indicate an image and represent each wavelet coefficient as a collection of ordered bits. The processor 112 codes the bits of each order to indicate zerotree roots that are associated with the order. Specification, p. 10, lines 6-17.

The codec 800 input 802, shown in Figure 8, includes frames of the incoming video sequence. The frames are coded by the codec 800 as intra (I), predicted (P) or skipped (S) frames. The I frame, which is sent at regular intervals starting from the first frame, contains the result of arithmetic coding (AC) 824 on the modified embedded zerotree (MEZT) coded 826 discrete wavelet transformed (DWT) image 828, as described previously herein. One suitable arithmetic coding technique is described in I.H. Witten et al., "Arithmetic Coding for Data Compression", Communications of the ACM, Vol. 30, No. 6, June 1987. Specification, p. 11, lines 14-21.

An error image, error frame or error data is the difference between two frames of image data. For the first frame, which is an intra or I-frame, MEZT may be applied as indicated at 826. The next frame is the predicted or P-frame, that is not directly encoded. Instead, the difference from the first or I frame is determined at 808 and that difference is encoded at 812. Specification, p. 11, lines 22-26.

The reconstructed predicted frame is the result of encoding error frames using reverse embedded zerotree coded error frames REZT 812 followed by inverse REZT (IREZT) 840. REZT will be explained later. A reconstructed image 830 is then developed from the motion estimation 804 to develop the skipped or S frame 832 and the predicted or P frame 806. The reconstructed frame 830 goes to the inverting input of summer 808. The error compensation 816 is developed from the error frame from IREZT 840 and added to the S frame at 832 to get the P frame 806. Specification, p. 11, line 27 through p. 12, line 2.

The programmable switch 814 is used to select an I frame, P frame or S frame based on quality feedback. If the error determined by the IREZT 840 is very small there is no need to use that data and so the data is simply skipped or dropped and only the motion vector 804 is transmitted which is received from the block 830. When the error is high, data is compensated or added at 813 and the P frame is used. The block 842 designates the selected frame as the previous frame for the motion estimation 804. Specification, p. 12, lines 3-8.

Regular transmission of I frames ensures robustness of the codec 800 against any channel error and removal of accumulated reconstruction error. However, since frequent transmission of I frames may reduce the compression ratio (CR), an optimization may be used to maintain a high CR as well as robustness of the codec 800. For the remaining frames, motion estimation 804 with respect to the previous reconstructed frames 806 is done using multi-resolution motion estimation (MRME) technique. Specification, p. 12, lines 9-14.

When the error accumulated by this process crosses a certain threshold as determined at 830, the codec 800 is partially refreshed by transmitting the stream generated by application of AC 812 on a REZT 812 error frame along with the motion vectors 832. Such frames, denoted by P, may be sent at an optimal frequency to maintain a high CR as well as a high peak signal to noise ratio (PSNR) of the reconstructed sequence. Specification, p. 12, lines 21-26.

Thus, switching between I, P and S frames is controlled by the energy of the error frame. The performance of the codec 800 may be primarily dependent on the efficiency of REZT 810 and hence also on the correctness and efficiency of MRME 804. Specification, p. 12, lines 27-29.

Usually, the codec 800 sends a motion vector m from 804 and optionally an error value e from 808. For the P frame both the motion vector m and the error value e are sent. For an S frame, only the motion vector m is sent. Specification, p. 13, lines 1-3.

The information generated by the processes described above is packed at 816, along with a header, to generate the output bit stream 818. The header 820 may contain information regarding the size of the frame, the length of the sequence along with the type of coding applied in each frame. Specification, p. 13, lines 4-7.

At the receiver end, shown in Figure 9, a reverse procedure may be followed using a sequence decoder 900. The first frame, being an I frame 902, is easily reconstructed by performing entropy decoding 904 followed by MEZT decoding 906. Specification, p. 13, lines 8-10.

For the other frames, the i th frame is predicted from the previously reconstructed frames (($i-1$)th) 920 and the transmitted motion vectors 922. In case of the P frame, the error frame 912 is added to the predicted frame 914 to complete the reconstruction process. The S frame is the motion vector without the error frame. Finally, inverse DWT (IDWT) 916 is applied to each of the frames to get the reconstructed sequence. Specification, p. 13, lines 11-15.

In one embodiment, the first block 828 of the codec 800 does the DWT operation, as shown in Figure 8. DWT results in decomposition of each of the input frames into a multi-resolution subband structure. Unlike the discrete cosine transform (DCT), discrete wavelet transformed images contain a lower resolution version of the original image that is usually called the low-frequency subband. Parameters such as the filter coefficients, the number of decomposition levels, etc. can be chosen depending upon the image sequence and its intended application. Moreover, for two-dimensional DWT, filtering is actually applied separately along

each dimension, which makes it parallelly realizable and hence suitable for real time applications. Specification, p. 13, lines 16-24.

In case of still image compression (i.e., the I frames where no prediction is applied), the matrix obtained after DWT is coded using MEZT scheme as indicated at block 826. This is an efficient bit-plane wise embedded zerotree coding scheme. Specification, p. 13, lines 25-27.

Since DWT is a multi-resolution transform, which generates a subband hierarchy, it is quite natural to use MRME to exploit this property. The resulting multi-resolution subband structure ensures a strong correlation between the motion activities of subbands at different positions and levels. Further, the blocking artifacts generated due to the simple transitory motion model are partially smoothened by the low pass filtering during inverse DWT. Specification, p. 13, line 28 through p. 14, line 2.

Wavelet transform decomposes a video frame into a set of sub-frames with different resolutions corresponding to different frequency bands. These multi-resolution frames provide a representation of the global motion vectors of the video sequence at different scales. Specification, p. 14, lines 3-6.

Although the motion activities for each sub-frame are not all identical, at the same time they are highly correlated and hence can be used as an excellent first approximation. In the MRME approach, motion vectors for higher resolution are derived from those of the lower resolution motion vectors. A variable block size approach is taken, in one embodiment, which not only reduces the search space and hence the computational time but also provides a meaningful characterization of the intrinsic motion structure (following the structure of the wavelet transformed sub-frames). Specification, p. 14, lines 7-13.

In one embodiment, the motion estimation 804 may be carried out using a three step search algorithm. But in principle, any other motion estimation technique may be suitable for applying in multiresolution hierarchical DWT subbands. The subsampled image (low-frequency subband) is first broken into blocks. Then for each block of a current frame (the frame for which motion estimation is being carried out) a matching block from a previous frame is identified using a distance criterion. The distance criterion may be a minimum mean square error, minimum mean of pixel by pixel absolute differences, or maximum matching pixel count as a few examples. Sum of Pixels by Pixel Absolute Difference (SAD) may be used as the distance criterion for choosing the best match. The frame is reconstructed at the transmitter end from the previous frame and the motion vectors for purposes of comparison. Specification, p. 14, lines 14-24.

In a three step search, SAD is calculated, at the center and at eight specific points within search window in the first step. The distances of these positions are four pixels away from center of the block of interest. The positions are the eight neighbors of a pixel. Depending upon the values of SAD at each search position, the next step search is carried out. At the next step, the search positions are along the same directions but around the position where minimum SAD was found at the previous step and the distance is reduced to two pixels instead of four pixels. The minimum SAD position is found in this step. The last step search is carried out around this position and now the distance is only one pixel. Here the window size becomes seven pixels $(4+2+1)$ along the x or y direction. So the search window is 15 X 15 pixels. Specification, p. 14, line 25 through p. 15, line 3.

For decoding, the block LL may be stored without any modification in one embodiment. For the blocks HL and LH the values are reconstructed using the associated motion vectors and

the previous frame. For the block HH, the average of the motion vectors for the corresponding HL and LH positions may be used to reconstruct from the previous frame. Specification, p. 15, lines 21-25.

The REZT 810 may significantly improve the efficiency of the codec 800 because it may generate a very highly compressed bit-stream for the error frame in DWT domain. The block 810 may result in a significant reduction in computational complexity by drastically reducing the number of scans necessary for coding the error frame coefficients. Specification, p. 15, lines 26-30. Specification, p. 15, lines 26-30.

The arithmetic coding 812 performs the task of entropy coding of the symbol stream generated by the REZT 810 for the error frames. A variety of arithmetic coding schemes may be used for entropy encoding. Specification, p. 16, lines 1-3.

The DWT-based video codec 800 may be computationally efficient because of reduced computational requirements in the multi-resolution motion estimation and bit-plane wise embedded zerotree coding schemes both for DWT frames and the error frames after motion estimation and compensation. The encoding scheme may work in one as opposed to two passes. This makes the codec 800 suitable for implementation both in software and hardware. Specification, p. 16, lines 4-9.

The REZT embedded coding scheme is suitable for error images or frames. The error image can be generated by taking difference of two successive DWT images in an image sequence. In the context of video, the error frame is the difference of the original frame from the reconstructed previous frame in the DWT domain. Motion prediction followed by motion compensation leads to generation of reconstructed frames. In error frames, the efficiency is increased by applying the embedded coding in HL, LH, and HH subbands only as shown in

Figure 10. The LL subband can be transmitted without any change. The embedded coding may be performed in every bit-plane in one embodiment. Specification, p. 16, lines 10-17.

The encoding algorithm 1100 for the n^{th} bit of a coefficient (c) in the error frame with respect to a threshold T_0 , shown in Figure 11, begins by determining whether the n^{th} bit is equal to zero, at diamond 1102. If not, a check at diamond 1104 determines whether the coefficient is less than zero. If so, the bit is labeled negative and if not, the bit is labeled positive.

Specification, p. 17, lines 11-15.

If the n^{th} bit is equal to zero as determined at diamond 1102, a check at diamond 1106 determines whether n^{th} bit of the coefficient forms a zerotree. If not, a check at diamond 1108 determines whether or not the absolute value of the coefficient is greater than the threshold T_0 . If so, the bit is labeled an isolated zero and if not it is labeled an absolute isolated zero.

Specification, p. 17, lines 16-20.

If the n^{th} bit of the coefficient forms a zerotree as determined at diamond 1106, a check at diamond 1110 determines whether the absolute value of the coefficient is less than T_0 . If not, the bit is labeled a root. If so, a check at diamond 1112 determines whether the magnitudes of all the descendants of the coefficient are less than T_0 . If so, the bit is labeled an absolute root and if not it is labeled a root isolated zero. Specification, p. 17, lines 21-25.

The error matrices (or frames) are discrete wavelet transformed (DWT) and the resultant subbands are of the form as shown in Figure 12. The labels (1,2,3) indicate the level number of the subbands. Specification, p. 17, lines 26-28.

The starting threshold is taken as 1 in one example. In each successive pass the threshold is doubled. The total number of such pass is $\lfloor \log_2(\max) \rfloor + 1$, where \max denotes the maximum

value among the magnitudes of all the coefficients to be encoded. The scanning pattern for the coefficients is shown in Figure 12. Specification, p. 17, line 29 through p. 18, line 2.

The parent-child relation for the DWT coded frames is shown in Figure 13. The four pixels at level 2 are the children of the pixel marked in level 1. The sixteen pixels at level 3 are also the descendants of marked pixel in the 1st level. Specification, p. 18, lines 3-5.

In each pass, whenever a coefficient is coded as ABS_ROOT, ABS_IZ or ROOT_IZ, the corresponding positions are suitably marked against further scanning. However, for ABS_IZ or ROOT_IZ, the marks are valid only for the ongoing pass. Specification, p. 18, lines 6-8.

This scheme may perform satisfactorily in case of error frames both for still imaging and video coding. In cases when the compression ratio is of more importance compared to PSNR, then pass_numbers (1,2,..) can be progressively dropped resulting in a significant increase in compression ratio (but incurring a loss in PSNR). If pass-levels are dropped progressively then the CR-performance of REZT improves at a better rate compared to that of MEZT. Specification, p. 23, lines 2-7.

The scheme may be computationally faster than the classical EZT technique. This scheme successfully avoids passing over the smaller or insignificant coefficients in every pass by encoding them in course of the initial passes. Further, the two passes of the classical scheme has been clubbed into a single pass. Specification, p. 23, lines 8-11.

Further compression can be achieved when the initial passes (1,2,..) are dropped progressively. The decoding scheme remains same as stated earlier with a minor modification: suitable number of zeros (depending on the number of passes dropped) are to be appended to the decoded data. However, such a process results in PSNR loss. Specification, p. 23, lines 12-15.

Other embodiments are within the scope of the following claims. For example, the matrices of decomposed coefficients described above have one coefficient in each subband of the highest decomposition level. However, this arrangement is for purposes of simplifying the discussion of the coding. Therefore, each subband of the highest decomposition level may have multiple coefficients, and the above-described techniques may be applied to code the bits associated with these coefficients. In some embodiments, the processor 112 may code all of the bits of each order in parallel. In this manner, the coding of the bits of each bit order may be performed by the processor's execution of a separate thread. Other arrangements are possible. Specification, p. 23, lines 16-24.

VI. ISSUES

- A. Is Claim 1 anticipated by the Van der Auwera reference?**
- B. Is Claim 2 rendered obvious over Van der Auwera and in view of the Lee reference?**

VII. GROUPING OF THE CLAIMS

For purposes of this appeal, claims 1, 3-10, 12-19 and, 21- 30 may be grouped together; the claims 2, 11, and 20 may be grouped in another group. Thus, the claims do not stand or fall together; the patentability of each group is discussed below.

VIII. ARGUMENT

- A. Is Claim 1 anticipated by the Van der Auwera reference?**

Claim 1 stands rejected under 35 U.S.C. § 102(e) over Van der Auwera. The method of claim 1 calls for providing error data to indicate motion in an image, determining a characteristic of the error data, and based on the characteristic, determining whether to use the error data to indicate motion in an image.

1. The choice must be made between use or non-use of error data based on a characteristic of the error data to indicate motion in an image.

As claimed in claim 1, use or non-use of error data based on the characteristic of error data is determined to indicate motion in an image. As such, the Van der Auwera reference does not teach that a choice is made between use or non-use of error data to indicate motion in an image based on a characteristic of the error data. That is, depending upon the type of coding of frame(s), motion compensated video frame is selectively used to generate error video frame which may be transmitted with motion vectors. However, Van der Auwera never teaches or suggests this choice of use or non-use of error data, as claimed in claim 1.

Van der Auwera instead of determining whether to use the error data based on the characteristic of error data, simply selects interframe coding or coding of each video frame separately for the motion compensated video frame(s). Thus, Van der Auwera fails to anticipate claim 1 limitations.

The Examiner appears to have disregarded this use and non-use selectivity of error data feature. However, this approach is incorrect because it essentially ignores specific limitations, of claim 1, distinguishing over the Van der Auwera reference. Accordingly, withdrawal of the § 102 rejection of claim 1 and the dependent claims therefrom is respectfully requested because claim 1 is patentably indistinguishable over the cited art. In this manner, Appellant respectfully submits that the claims depending from claim 1 are also in condition for allowance because they depend from an allowable claim.

2. The rejection is based on a notion that the motion vectors only play a role in case interframe coding is selected.

As claimed in claim 1, error data is provided to indicate motion in an image (regardless of type of a frame). However, in the Van der Auwera reference, there is no teaching whatsoever as

to providing error data to indicate motion in an image regardless of a frame type. Instead, Van der Auwera limits the use of motion vectors to the case in which interframe coding is selected.

not C.
In fact, when considering Van der Auwera disclosure in relation to independent claim 1, there is no teaching of (use of motion vectors for all types of frame coding.) That is, motion vectors (for example, for motion compensation circuit 80) only play a role in case interframe coding is selected within the interface encoding circuit (110). In contrast, claim 1 includes the limitation that error data is provided to indicate motion in an image and a characteristic of the error data is determined, and based on the characteristic, selective use of the error data to indicate motion in an image is determined. Therefore, instead of using error data to indicate motion in an image, Van der Auwera simply selects interframe coding or coding of each video frame separately for the motion compensated video frame(s). Accordingly, Applicant respectfully requests reversal of the Examiner's rejection of claim 1 and claims depending therefrom.

In other words, in claim 1, a specific judgment is made as to use of the error data.

✓ (However, no suggestion of use or non-use of error data is indicated in the Van der Auwera reference. In the Office Action mailed November 7, 2003, on page 4, the Examiner contends that claim 1 is rejected for the same reasons as set forth in Section 4 of the previous Office Action, Paper No. 4, dated July 7, 2003. However, the Van der Auwera reference simply discloses providing error data as a selective adder or subtracter. The circuit 20 in the cited reference is not error data but it is a block that does addition or subtraction. It is suggested that the motion vector 130 is used when the magnitude of the error data falls below a particular value. However, a motion estimation circuit 70 produces motion vectors 190 for the motion compensation circuit 80. The motion vector 130 and encoded video frame 120 are transmitted.

not C.
There is no discussion or teaching as to (use of magnitude of some error data) such that error data

may be selectively used, as included in claim 1. As to the multi-resolution motion estimation (90, 100), the motion estimation circuit simply determines one of estimation error norms.

✓ (However, there is no suggestion or teaching for determining based on a characteristic of error data whether to use the error data to indicate motion of an image. In short, there is no support for the rejection within the Van der Auwera reference.

B. Is Claim 2 rendered obvious over Van der Auwera and in view of Lee references?

Claim 2 limitations include representing error data as a collection of ordered bits, and coding the bits of each order to indicate zerotree roots that are associated with the order. In this arrangement, unlike conventional zerotree coding schemes, thresholds are not computed to identify insignificant values as the “0” bit is treated as being insignificant and the “-1” and “1” bits are treated as being significant. In this manner, instead of classifying the wavelet coefficients (as zerotree roots, or isolated zeros, as examples), codes to classify the bits of the wavelet coefficients are produced. See Applicant’s specification on page 4, lines 28-31.

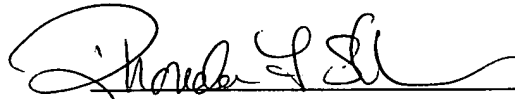
However, the Lee reference fails to teach any of these steps, let alone the specific limitations included in claim 2. The Lee reference simply teaches a wavelet tree generator 204 that performs a wavelet hierarchical subband decomposition to produce a conventional wavelet tree representation of an input image. See column 4, lines 35-48. In other words, error data is not represented as a collection of ordered bits and coding of the bits of each order is not indicated as zerotree roots that are associated with the order. In view of these remarks, the application should now be in condition for allowance.

IX. CONCLUSION

The Applicant's respectfully request that each of the final rejections be reversed and that the claims subject to this appeal be allowed to issue.

Respectfully submitted,

Date: May 13, 2004



Rhonda L. Sheldon
Registration No. 50,457
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, Texas 77024-1805
713/468-8880 [Phone]
713/468-8883 [Fax]

Customer No. 21906

APPENDIX OF CLAIMS

- 1 1. A method comprising:
2 providing error data to indicate motion in an image;
3 determining a characteristic of the error data; and
4 based on said characteristic, determining whether to use said error data to
5 indicate motion in an image.

- 1 2. The method of claim 1 including representing error data as a collection of
2 ordered bits, and coding the bits of each order to indicate zerotree roots that are
3 associated with the order.

- 1 3. The method of claim 1 wherein providing error data includes taking the
2 difference between two successive image representations in an image sequence.

- 1 4. The method of claim 3 wherein taking the difference includes taking the
2 difference of two successive discrete wavelet transform coded frames.

- 1 5. The method of claim 1 wherein determining a characteristic includes
2 determining whether or not the error data exceed a predetermined threshold value.

- 1 6. The method of claim 5 including, if the magnitude of the error data is
2 below the value, using a motion vector to indicate motion in the image.

1 7. The method of claim 5 wherein if the magnitude of the error data exceeds
2 the value, using a motion vector and the error data to indicate motion in an image.

1 8. The method of claim 5 including zerotree encoding said error data.

1 9. The method of claim 8 including zerotree encoding a representation of the
2 intensity values of pixels making up an image.

1 10. An article comprising a storage medium readable by a processor-based
2 system, the storage medium storing instructions to enable a processor to:
3 provide error data to indicate motion in an image;
4 determine a characteristic of the error data; and
5 based on said characteristic, determine whether to use said error data to
6 indicate motion in an image.

1 11. The article of claim 10, the storage medium comprising instructions to
2 enable the processor to:
3 represent error data as a collection of ordered bits and code the bits of each
4 order to indicate zerotree roots that are associated with the order.

1 12. The article of claim 10, the storage medium comprising instructions to
2 enable the processor to take the difference between two successive image representations
3 in an image sequence to develop the error data.

1 13. The article of claim 12, the storage medium comprising instructions to
2 enable the processor to take the difference of two successive discrete wavelet transform
3 coded frames.

1 14. The article of claim 10, the storage medium comprising instructions to
2 enable the processor to determine whether or not the error data exceed a predetermined
3 threshold value.

1 15. The article of claim 14, the storage medium comprising instructions to
2 enable the processor to, if the magnitude of the error data is below the value, use a
3 motion vector to indicate motion in the image.

1 16. The article of claim 14, the storage medium comprising instructions to
2 enable the processor to, if the magnitude of the error data exceeds the value, use a motion
3 vector and the error data to indicate motion in the image.

1 17. The article of claim 14, the storage medium comprising instructions to
2 enable the processor to zerotree encode said error data.

1 18. The article of claim 17, the storage medium comprising instructions to
2 enable the processor to zerotree encode a representation of the intensity values of pixels
3 making up an image.

1 19. A system comprising:
2 a subtracter to provide error data to indicate motion in an image; and
3 a device to determine a characteristic of the error data and, based on the
4 characteristic, determine whether to use the error data to indicate motion in an image.

1 20. The system of claim 19 wherein said device represents error data as a
2 collection of ordered bits, and codes the bits of each order to indicate zerotree roots that
3 are associated with the order.

1 21. The system of claim 19 wherein the subtracter takes the difference
2 between two successive image representations in an image sequence to develop error
3 data.

1 22. The system of claim 21, wherein the subtracter takes the difference of two
2 successive discrete wavelet transform coded frames.

1 23. The system of claim 19 wherein the device determines whether or not the
2 error data exceeds a predetermined threshold value.

1 24. The system of claim 23 wherein the device uses a motion vector only to
2 indicate motion in the image if the magnitude of the error is below the value.

1 25. The system of claim 23 wherein the device uses a motion vector and error
2 data to indicate motion in the image if the magnitude of the error exceeds the value.

1 26. The system of claim 23 wherein said device zerotree encodes said error
2 data.

1 27. The system of claim 26 wherein the device zerotree encodes a
2 representation of intensity values of pixels making up an image.

1 28. The system of claim 19 including arithmetic coder to code said error data.

1 29. The system of claim 19 wherein said device zerotree encodes said error
2 data and inverts the zerotree encoding of said error data.

1 30. The system of claim 19 wherein said device uses multi-resolution motion
2 estimation.